

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平6-103064

(43)公開日 平成6年(1994)4月15日

(51)Int.Cl. <sup>5</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 9/34 9/315 9/355	3 3 0	9189-5B 9189-5B	G 0 6 F 9/ 30 9/ 36	3 4 0 D 3 2 0
審査請求 未請求 請求項の数11(全 22 頁)				

(21)出願番号 特願平4-276665

(22)出願日 平成4年(1992)9月21日

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72)発明者 中原 茂

東京都青梅市今井2326番地 株式会社日立

製作所デバイス開発センタ内

(74)代理人 弁理士 玉村 静世

(54)【発明の名称】 データ処理装置及びそのデータ処理方法

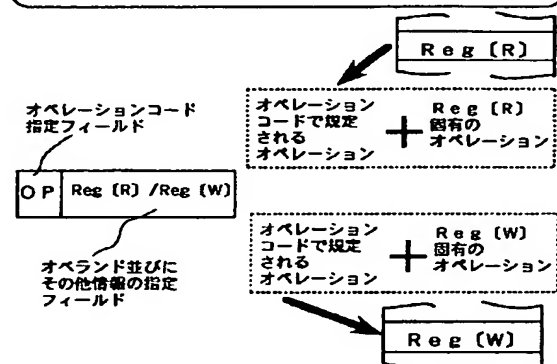
(57)【要約】

【目的】 オペレーションコードのビット数や命令制御部の論理規模の増大を抑えつつ機能を多様に拡張できるデータ処理装置を提供することにある。

【構成】 レジスタに対するデータの書き込みまたは読み出し時に、そのデータに対して特定の処理を施す機能を予め所定のレジスタ（機能付きレジスタ）Reg [R] , Reg [W] に割当て、その機能付きレジスタを命令中にて指定することにより、そのデータに対して、命令のオペレーションコードによって規定される命令の処理と機能付きレジスタの有する特定機能とを組み合わせた多様なデータ処理を実現するものである。

【図1】

- (1) Operating read data Register  
if read data is read from Reg[R]  
{ read data ← Fr (i) (Reg[R]) } and/or { Reg[R] ← Fr (i) (Reg[R]) };  
Fr (i) : i番目の命令の特定フィールドの値をパラメータとする  
レジスタR固有の演算子
- (2) Operating write data Register  
if write data is written to Reg[W]  
Reg[W] ← Fw (i) (Reg[W]);  
Fw (i) : i番目の命令の特定フィールドの値をパラメータとする  
レジスタW固有の演算子



【特許請求の範囲】

【請求項1】 命令制御部の制御に基づいて命令を実行部で実行するデータ処理装置において、前記実行部に単数若しくは複数の機能付きレジスタを設けたものであって、

前記機能付きレジスタは、命令中でその利用が指定され、それを指定する命令のオペレーションコードで指示される処理によって当該機能付きレジスタから読出された情報或は書き込まれるべき情報に対して、当該オペレーションコードで指定される処理とは別の特定の処理を付加する機能が予め割当てられたレジスタであることを特徴とするデータ処理装置。

【請求項2】 前記機能付きレジスタは、これに書き込むべきデータの各ビット値を反転して保持するノットレジスタであり、当該ノットレジスタの入力には反転回路が接続されて成るものである請求項1記載のデータ処理装置。

【請求項3】 前記機能付きレジスタは、それが保持する値の読み出しが指定されることによって、当該読み出しデータに所定の定数を加算した値を再度保持するインクリメントレジスタであり、前記加算すべき定数を選択して出力する選択回路と、選択回路の出力と前記インクリメントレジスタの出力を加算して当該インクリメントレジスタの入力に供給する加算器とを備えて成るものである請求項1記載のデータ処理装置。

【請求項4】 前記機能付きレジスタは、これを指定する命令の偏値が負ならば保持値をリードデータとして前記命令による処理に引渡し、また、偏値が0または正ならば保持値から偏値を引いたものをリードデータとして前記命令による処理に引渡し、且つ、リードデータを引渡しした後当該保持値に前記偏値を加算した値を次の保持値として更新するモディファイドレジスタである請求項1記載のデータ処理装置。

【請求項5】 前記機能付きレジスタは、書き込みデータに対してマスクパターンとの論理積が採られたものを保持するアライメントレジスタである請求項1記載のデータ処理装置。

【請求項6】 機能付きレジスタは、書き込むべきデータにプログラムステータスワードのキャリー又はボロウの状態を表わすビットの値を加算したものを保持するキャリー/ボロウレジスタである請求項1記載のデータ処理装置。

【請求項7】 前記機能付きレジスタは、保持値の値を所定ビットだけ所定方向にシフトさせた値を、そのリードデータとして命令の処理の引渡しシフトレジスタである請求項1記載のデータ処理装置。

【請求項8】 命令の記述に従って処理を行うデータ処理方法において、  
前記命令の記述は、オペレーションコードと、オペレーションコードで指定された処理に利用すべきレジスタの

指定情報とを含み、

その命令中で指定されたレジスタに予め割当てられている固有のオペレーションを、当該レジスタの保持値に対して行い、その結果を前記レジスタからの読み出しデータとしてオペレーションコードで指定された処理に引渡す処理を行うことを特徴とするデータ処理方法。

【請求項9】 前記オペレーションコードで指定された処理に引渡した後、その引渡したリードデータを当該レジスタに書き戻す処理を更に含む請求項8記載のデータ処理方法。

【請求項10】 命令の記述にしたがって処理を行うデータ処理方法において、

前記命令の記述は、オペレーションコードと、オペレーションコードで指定された処理に利用すべきレジスタの指定情報とを含み、

その命令中で指定されたレジスタに予め割当てられている固有のオペレーションを、その命令のオペレーションコードによって当該レジスタへの書き込みが指定されたデータに対して行い、これを当該レジスタに書き込む処理を行うことを特徴とするデータ処理方法。

【請求項11】 前記命令は、予め固有のオペレーションが割当てられている前記レジスタとそれ以外の汎用レジスタとを区別可能なビットフィールドを前記レジスタ指定情報のためのフィールドに有するものである請求項9又は10記載のデータ処理方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、オペレーションコードの増大を抑えつつ高機能な処理を実現し、また高機能な処理を少ないステップ数を以て実現可能なデータ処理装置並びにデータ処理方法に係り、例えば、マイクロコンピュータのような論理集積回路装置に適用して有効な技術に関するものである。

【0002】

【従来の技術】フォン・イノマン型などのデータ処理装置の基本的な構成は程んど同じであり、例えば、キャッシュメモリ等に記憶した一連の命令を順番に命令制御部に読み込み、その命令が何であるかを表わしているオペレーションコードと呼ばれるビット領域をデコードしてデータに施すべき処理を決定し、更にオペランドと呼ばれる被演算データを指定する領域をデコードしてデータを取り込んで、上述した処理をそのデータに対して行うものである。このようなデータ処理装置の一例としては、Mips（ミップス）社のR4000チップ（日経エレクトロニクス、1991年10月14日経PB社発行、No. 496号）、Intel（インテル）社のi960チップ（日経エレクトロニクス、1990年1月8日経PB社発行、No. 490号）、及びHP（ヒューレットパッカード）社のPA-RISCチップ（日経マイクロデバイス、日経PB社発行の1990年

9月号)などを挙げることができる。

【0003】また、データ処理装置のアーキテクチャは、比較的簡単な命令セットを以って処理の高速化並びにハードウェアの簡素化を図ろうとするRISC (Reduced Instruction Set Computer) 的アーキテクチャと、オブジェクト指向のアーキテクチャのような比較的複雑な命令セットを以って高機能化を目指すCISC (Complex Instruction Set Computer) 的アーキテクチャに大別できる。現実に提供されている各種データ処理装置のアーキテクチャがRISCかCISCの何れであるかというような峻別は実際には難しいが、多様なアドレッシングモードや、一つの命令で複数の演算を行う複合命令などをCISC的な命令と考えることができる。

【0004】

【発明が解決しようとする課題】しかしながら、前記RISC的なデータ処理装置において、CISC的なデータ処理装置が1命令で提供するような高機能な処理は、基本的に簡単な処理を実現する命令を複数組み合わせで対処していた。このため、出現頻度は少ないかもしれないが、そのような処理を能率的に行うことができなかった。このとき、多様なアドレッシングモードや、一つの命令で複数の演算を行う複合命令など、CISC的な命令を基本的な命令セットに追加しようとすると、ハードウェアがサポートする命令数が増加してしまう。その結果、命令制御部のデコーダ等のランダムロジックの論理規模が大きくなり、クリティカルパスの遅延時間の増大、すなわちLSIチップの性能低下やチップ面積の増大、開発日程の増大化などを招いてしまっていた。

【0005】本発明の目的は、オペレーションコードの増大を抑えつつ機能の多様化を実現し、また高機能な処理を少ないステップ数を以って実現可能なデータ処理装置並びにそのデータ処理方法を提供することにある。今発明の別の目的は、ハードウェアの増大を極力抑えて多様な機能を実現できるデータ処理装置を提供することにある。

【0006】本発明の前記並びにその他の目的と新規な特徴は本明細書の記述及び添付図面から明らかになるであろう。

【0007】

【課題を解決するための手段】本願において開示される発明のうち代表的なものの概要を簡単に説明すれば下記の通りである。

【0008】すなわち、レジスタに対するデータの書き込みまたは読み出し時に、そのデータに対して特定の処理を施す機能を予め所定のレジスタ (機能付きレジスタ) に割当て、その機能付きレジスタを命令中に指定することにより、そのデータに対して、命令のオペレーションコードによって規定される命令の処理と機能付きレ

ジスタの有する特定機能とを組み合わせた多様なデータ処理を実現するものである。

【0009】

【作用】上記した手段によれば、命令のオペレーションコードによって規定される命令の処理と機能付きレジスタの有する特定機能とを組み合わせることで多様なデータ処理を実現できる。このことは、命令セットの命令数を増加することなしに、換言すれば、オペレーションコードのビット数を増大させることなく、データ処理装置による処理を多様化若しくは多機能化するように作用し、オペレーションコードのデコード論理に代表されるような命令制御部のハードウェアの増大を極力抑えて多様な機能を実現できる。

【0010】上記によって実現される多様な機能は一つの命令中のオペランド指定領域などでの機能付きレジスタの指定で行われ、且つ、その機能付きレジスタに割当てられる固有の機能は当該レジスタに対するデータの書き込みまたは読み出し時に行われるので、機能拡張に伴って実行すべき命令数が増えず、このことが、高機能な処理を少ないステップ数を以って実現するように作用する。

【0011】

【実施例】図1には本発明に係るデータ処理装置の機能付きレジスタ (以下オペレーティングレジスタとも記す) の定義と機能の一例が示される。同図におけるオペレーティングレジスタの一般的定義はC言語的な形態で示され、オペレーティングレジスタは、(1)に示されるオペレーティング・リード・データ・レジスタ (Operating Read Data Register) Reg [R] と、(2)に示されるオペレーティング・ライト・データ・レジスタ (Operating Write Data Register) Reg

[W] に大別される。オペレーティング・リード・データ・レジスタ Reg [R] は、予じめ定義されている Reg [R] 固有のオペレーション Fr (i) を、i 番目の命令の Reg [R] で規定される特定のフィールドの値をパラメータとして、Reg [R] の内容に対して行い、その結果を読み出しデータ (read data) とするか、再び当該 Reg [R] に書き戻すかのどちらか一方か、またはその両方を行う。

【0012】オペレーティング・ライト・データ・レジスタ Reg [W] は、予じめ定義されている Reg [W] 固有のオペレーション Fw (i) を、i 番目の命令の Reg [W] で規定される特定のフィールドの値をパラメータとして、書き込みデータ (write data) に対して行い、当該 Reg [W] に書き込む。

【0013】前記オペレーティングレジスタ Reg [R]、Reg [W] は、図1に示されるようなフォーマットの命令の中で指定することができる。図に示される命令は、オペレーションコード指定フィールドとオペ

ランド並びにその他情報の指定フィールドを有し、オペレーションコード指定フィールドには、データ処理装置のアーキテクチャに従って用意されているオペレーションコード（LOADやADDといった命令コード）OPが記述され、オペランド並びにその他情報の指定フィールドには、前記オペレーションコードに従って命令を実行するために必要なオペランド（演算対象となるもの）それ自体やその所在、更にはその他制御情報が記述される。前記オペレーティングレジスタReg [R]、Reg [W]は、命令フォーマット中において前記オペランド並びにその他情報の指定フィールドで指定される。その指定は、例えば、オペレーティングレジスタReg [R]、Reg [W]に割当てられた固有のレジスタ番号で指定することができる。

【0014】このように固有の機能が割当てられたオペレーティング・リード・レジスタReg [R]を命令中で指定したとき、当該命令の実行においては、その命令のオペレーションコードで規定される処理（オペレーション）とオペレーティング・リード・データ・レジスタReg [R]固有の処理が、当該オペレーティング・リード・データ・レジスタReg [R]の保持情報を利用して行われる。例えば、その命令中で指定されたオペレーティングレジスタReg [R]に予め割当てられている固有のオペレーションを、当該レジスタの保持値に対して行い、その結果を前記オペレーティングレジスタReg [R]からの読み出しデータとしてオペレーションコードで指定された処理に引渡す処理を行う。前記オペレーションコードで指定された処理に引渡した後、その引渡した読み出しデータを当該オペレーティングレジスタReg [R]に書き戻す処理を必要に応じて更に行うことができる。

【0015】また、固有の機能が割当てられたオペレーティング・ライト・データ・レジスタReg [W]を命令中で指定したとき、当該命令の実行においては、その命令のオペレーションコードで規定される処理とオペレーティング・ライト・データ・レジスタReg [W]固有の処理が行われ、その結果が当該オペレーティング・ライト・データ・レジスタReg [W]に格納される。例えば、その命令中で指定されたオペレーティングレジスタReg [W]に予め割当てられている固有のオペレーションを、その命令のオペレーションコードによって当該オペレーティングレジスタReg [W]への書き込みが指定されたデータに対して行い、これを当該オペレーティングレジスタReg [W]に書き込む処理を行う。

【0016】オペレーティング・リード・データ・レジスタReg [R]と、オペレーティング・ライト・データ・レジスタReg [W]に割当てられた夫々固有の処理は、当該レジスタに対するデータの読み出し又は書き込みの際に行われる。

【0017】この原理的な説明から明らかなように、既存の所定オペレーションコードにオペレーティングレジスタReg [R]、Reg [W]の指定を伴うことにより、オペレーションコードの種類を増やすことなく同一オペレーションコードすなわち1命令によって実現可能な処理を多様化若しくは豊富化でき、命令数並びに命令制御部の論理規模を増やすことなく高機能的な命令と同様の処理をサポートできるようになる。見方を変えれば、もともと高機能的な命令を専用のオペレーションコードで用意しているデータ処理装置においては、その機能を削減することなく命令数並びに命令制御部の論理規模を減らすことができる。

【0018】次に図2から図8を参照しながら前記オペレーティングレジスタの具体例、並びにそれを用いた処理の一例をメモリ・レジスタ間のデータの流れに着目して概念的に説明する。

【0019】図2にはノット・レジスタのC言語的な定義並びにそれを用いた処理例が概念的に示される。ノットレジスタNは、前記オペレーティング・ライト・データ・レジスタReg [W]の範疇に含まれ、ノットレジスタN (Reg [N])への書き込みデータの各ビット値を反転させてノットレジスタN (Reg [N])へ書き込むものである。同図には応用例としてディスティネーションにノット・レジスタNを指定したLOAD命令として「LOAD disp(b), N」が示されている。この命令は、ディスティネーションにノットレジスタNを指定することにより、汎用レジスタb {GR [b]}の値（ベースアドレス）に偏値（disp）を加算したメモリアドレスに格納されているデータをノットレジスタNに書き込むとき、当該ノットレジスタ固有の機能としてそのデータをビット反転する機能が付加される。したがって、同図に示される命令「LOAD disp(b), N」のように、そのディスティネーションにノットレジスタNを指定することにより、ノットレジスタNへの書き込みデータをノットレジスタNへ書き込む処理と共にその書き込みデータのビット反転を1命令（LOAD命令）で実行することができる。

【0020】図3にはインクリメントレジスタのC言語的な定義並びにそれを用いた処理例が概念的に示される。インクリメントレジスタIn (Reg [In])は、前記オペレーティング・リード・データ・レジスタReg [R]の範疇に含まれ、インクリメントレジスタIn (Reg [In])のデータの読み出しが行われる際に、ある定数だけ、そのデータが示す内容に増加させて再びインクリメントレジスタIn (Reg [In])に書き戻すものである。同図の応用例では、インクリメントレジスタInはLOAD命令のベースアドレス指定用のレジスタとして使用される。このような応用例においては、一つのデータをメモリからレジスタに書き込みをする毎にインクリメントレジスタInのベースアドレ

スが自動的に定数ずつ増加されるので、連続するメモリ領域のデータの読み出し処理を、通常のLOAD命令を用いて実行できる。

【0021】図4にはモディファイドレジスタのC言語的な定義並びにそれを用いた処理例が概念的に示される。モディファイドレジスタM (Reg [M]) は、前記オペレーティング・リード・データ・レジスタReg [R] の範疇に含まれ、i 番目の命令の偏値 (disp) が負ならばモディファイドレジスタM (Reg [M]) が保持する値を読み出しデータ (read data) とし、偏値 (disp) が0または正ならばモディファイドレジスタM (Reg [M]) の値から偏値を引いたものを読み出しデータ (read data) とする。また、読み出しが行われる毎に、モディファイドレジスタM (Reg [M]) の内容に偏値を加算したデータを再びモディファイドレジスタM (Reg [M]) に書き戻す。このモディファイドレジスタM (Reg [M]) をLOAD命令におけるベースアドレス指定用のレジスタとして使用した応用例が同図に示される。この例は、偏値 (disp) が負の場合であり、モディファイドレジスタMの定義における読み出しデータはモディファイドレジスタMの値であるから、このときのメモリアクセスアドレスは、モディファイドレジスタMの値に偏値 (disp) を加算した値 (Reg [M] + disp) とされる。応用例における偏値 (disp) が正の場合の具体例は図示していないが、モディファイドレジスタMの定義における読み出しデータはモディファイドレジスタMの値から偏値 (disp) を引いたものとされるから、このときのメモリアクセスアドレスは、モディファイドレジスタMの値に等しい値 (Reg [M] - disp + disp) になる。これにより、スタック若しくはFIFO (First-in First-out) のプッシュ/ポップのためのポストインクリメント (Post-Increment) とプリデクリメント (Pre-Decrement) のアドレッシングを、通常のLOAD命令により実現することができるようになる。

【0022】図5にはモディファイドインデックスレジスタのC言語的な定義が示される。モディファイドインデックスレジスタMi (Reg [Mi]) は、図3のインクリメントレジスタIn (Reg [In]) のオペレーションで、定数 (Const) の代わりにi 番目の命令で指定されたレジスタReg [r] の値を使うものである。図3の応用例にモディファイドインデックスレジスタMi (Reg [Mi]) を用いる場合、予じめReg [r] に適当な値を入れておくことにより、任意の値でベースアドレスの増加が行える。

【0023】図5には更にアライメントレジスタのC言語的な定義並びにそれを用いた処理例が概念的に示される。アライメントレジスタAL (Reg [AL]) は、

前記オペレーティング・ライト・データ・レジスタReg [W] の範疇に含まれ、インデックスで指定されるマスクパターンとアライメントレジスタAL (Reg [AL]) への書き込みデータの各ビットごととの論理積を採るものであり、その論理積の処理をした後のデータをアライメントレジスタAL (Reg [AL]) へ書き込む。例えば応用例1のようにマスクパターン (AL0) が上位ハーフワードは全て0、下位ハーフワードが全て1と定義されているものとする、LOAD命令のディスティネーションとして指定されたアライメントレジスタAL0には、下位ハーフワードにはメモリの対応するビットの値が、上位ハーフワードには全ビット0のデータが書き込まれる。このアライメントレジスタAL0は図6に示される応用例2のように、ディスティネーションのレジスタtにデータを書き込む際の中間バッファとして使用することもできる。図6においてその中間バッファはAg0、マスクパターンは(Ag0)として図示されている。したがって、応用例1、2に示すように、アライメントレジスタALを書き込みデータのディスティネーション (又は中間バッファ) として指定することにより、その書き込みデータとマスクパターンとの論理積の処理とアライメントレジスタAL (レジスタt) への書き込み処理とが1命令 (LOAD命令) で実行できる。

【0024】図7にはキャリー/ボロー (Carry/Borrow) レジスタのC言語的な定義並びにそれを用いた処理例が概念的に示される。キャリー/ボローレジスタC/B (Reg [C/B]) は、前記オペレーティング・ライト・データ・レジスタReg [W] の範疇に含まれ、キャリー/ボローレジスタC/B (Reg [C/B]) への書き込みデータ (write data) にプログラムステータスワードPSW (Program Status Word) のキャリーやボローの状態を表わすビット、すなわちPSW [C/B] の値を加算してキャリー/ボローレジスタC/B (Reg [C/B]) へ書き込むものである。応用例のように加減算命令 (ADD命令等) のディスティネーションをキャリー/ボローレジスタC/Bに指定することにより、1命令の実行で、二つのレジスタa、bの内容の加減算結果に、さらに前回の演算で発生したキャリーもしくはボローの値PSW [C/B] を加えることができる。

【0025】図8にはシフトレジスタReg [ni] のC言語的な定義並びにそれを用いた処理例が概念的に示される。シフトレジスタni (Reg [ni]) は、前記オペレーティング・リード・データ・レジスタReg [R] の範疇に含まれ、シフトレジスタni (Reg [ni]) の内容をiビットだけ左側にシフトし、読み出しデータ (read data) として出力する。同図の応用例では、ADD命令が示されていて、この命令を実行すると、シフトレジスタn2

から読み出されたデータは2ビット左シフトされた後にレジスタbの値と加算され、その加算結果のデータがディスティネーションのレジスタtに書き込まれる。

【0026】図9には図2から図8で説明した機能付きレジスタを用いたデータ処理装置の一実施例ブロック図が示される。同図に示されるデータ処理装置は、公知の半導体集積回路製造技術によってシリコンのような1個の半導体基板に形成され、命令制御部1、演算部2、命令キャッシュメモリ3、及びデータキャッシュメモリ4が代表的に示される。

【0027】同図の演算部2に示されるE3は、図2から図8で説明した各種機能付きレジスタの集合（以下単に機能付きレジスタとも記す）である。演算部2において、機能付きレジスタE3の機能実現に専用化された回路ブロックとして、特に制限されないが、セクタE1、加算器E2、及び反転回路E4が新たに設けられる。マスク回路E5、セクタE6、及びプリシフトE7は、データ処理装置には通常設けられていると共に前記機能付きレジスタE3の機能実現にも利用される回路ブロックである。演算部2にはその他に、複数本のレジスタから成る汎用レジスタE8、プログラムカウンタ（PC）E10、算術論理演算器（ALU）E9、及び前回の演算で発生したキャリー又はボローの値PSW〔C/B〕等を保持しているプロセッサステータスワードレジスタ（PSW）E11などが設けられている。B1、B2、B3、B4は演算部2において代表的に示された内部バスであり、DV0～DV9はバイパス用ドライバである。また、機能レジスタ内のアライメントレジスタは、AL0の他にAL1があり、それぞれのアライメントレジスタに対応してマスク回路のマスクパターンが後述する制御信号S6により制御される。

【0028】命令制御部1は命令キャッシュメモリ3からフェッチした命令を解読して、代表的に図示された各種の制御信号S1乃至S17などの制御信号を所定のタイミングを以て演算部2などに供給する。命令制御部1には、オペランドデコーダI1、オペコードデコーダI2、ターゲットデコーダI3、及び制御ブロックI4、I5が含まれる。オペコードデコーダI2は命令に含まれるオペレーションコードを解読する。命令のその他のフィールドは、特に制限されないが、オペレーションコードの種類に従って、オペランドデコーダI1やターゲットデコーダI3に供給される。オペランドデコーダI1は例えば命令のオペランド若しくはソース指定領域を解読する。この領域に機能付きレジスタとしての前記オペレーション・リード・データレジスタReg

〔R〕が指定されている場合には当該レジスタの指定が解読される。ターゲットデコーダI3は例えば命令のディスティネーション若しくはターゲット指定領域を解読する。この領域に機能付きレジスタとしての前記オペレーション・ライト・データレジスタReg〔W〕が指定

されている場合には当該レジスタの指定が解読される。オペランドデコーダI1及びターゲットデコーダI3による解読結果は、汎用レジスタE8や機能付きレジスタE3の指定に利用される。また、オペランドデコーダI1及びターゲットデコーダI3による解読結果は、オペコードデコーダの解読結果と共に制御ブロックI4にも供給されて、セクタE1、マスク回路E5、プリシフトE7の制御並びにキャリー／ボロー制御、そして、バイパス用ドライバDV0～DV9の開閉制御などに利用される。

【0029】ここで図10を参照しながら汎用レジスタE8や機能付きレジスタE3の指定手法について説明する。双方のレジスタは共に、命令中におけるオペランド並びにその他の情報のための指定フィールドで指定される。実際に命令中のどの領域すなわち第何ビット目から第何ビット目迄を利用するかはオペレーションコードの種類によって予め規定されている。例えば図10に示されるようにLAOD命令が、オペコード（オペレーションコード）、オペランド、ディスティネーション、偏値（disp）の各フィールドを有するとき、オペランド又はディスティネーションの領域でレジスタが指定される。例えば汎用レジスタE8と機能付きレジスタE3に含まれるレジスタが全部で2<sup>n</sup>個あるときには、nビットの情報によって順番にレジスタ番号を指定することができる。また、図10に示されるように、例えば、汎用レジスタE8が8個の汎用レジスタGEReg1～GEReg8を含み、機能付きレジスタE3が7個の機能付きレジスタOPReg1～OPReg7を含むとき、個々のレジスタには同図に示される4ビットB3、B2、B1、B0で規定される番号を割り当てることができる。このように規定した場合には、レジスタ指定情報B3～B0の最上位ビットB3の“0”は汎用レジスタE8を指定するビットとみなされ、レジスタ指定情報の最上位ビットB3の“1”は機能付きレジスタE3を指定するビットとみなされる。したがって、このようにレジスタ番号の割り付けを行った場合、命令フォーマット中のレジスタ指定領域には汎用レジスタを指定するのか或は機能付きレジスタを指定するのかを指定するビットフィールド（B3）が存在することになる。

【0030】プログラムカウンタE10で指定されたアドレスにある命令は、命令キャッシュメモリ3から読み出され、命令制御部1へ送られる。命令制御部1は命令中のオペコード及びオペランドなどをデコードし、そのデコード結果にしたがった各種制御信号などで演算部2の動作を制御する。

【0031】図11及び図12には図2から図8に示される各種機能付きレジスタに割当てられているオペレーションを図9の構成で実行するときの制御形態がパイプライン形式で示される。同図に示されるパイプライン処理は、命令フェッチ、命令デコード、演算、メモリアク

セス、及びレジスタライト／メモリストアの5段のバイプラインステージから構成され、機能付きレジスタを指定した処理毎に、どのステージでどのような処理が行われるかが示されている。同図において実線矢印はデータの流れ、破線矢印はアサートされた制御信号又はアドレス信号の流れを意味する。そして信号に付された符号並びにブロック内の符号は図9に示される符号に対応している。図2乃至図8で説明した各種機能付きレジスタに割当てられているオペレーションの一例を図9の構成に即して説明していく。

【0032】LOAD命令等の実行時に、命令デコードステージにおいて、ターゲットデコーダ13が、デコードしたレジスタ番号がノットレジスタNの場合は、マスク回路E5を介して読み出されるデータキャッシュメモリ4の読み出しデータ又は内部バスB2（以下書き込みバスとも記す）に読み出されているデータのビット値が、レジスタライトメモリストアステージでビット値を反転させる反転回路E4により、反転させられレジスタ選択制御信号S7により選択されるノットレジスタNに書き込まれる〔図11（1）、（2）〕。

【0033】LOAD命令等の実行時に、命令デコードステージにおいて、ターゲットデコーダ13が、デコードしたレジスタ番号がアライメントレジスタAL0又はAL1の場合は、ターゲットデコーダ13のデコード値により指定されたアライメントレジスタAL0又はAL1に対応するマスク回路E5のマスクパターンが、上記デコード値に基づいて、制御ブロック14から出力される制御信号S6により選択される〔図12（11）〕。そして、レジスタライトメモリストアステージにおいて、データキャッシュメモリ4の読み出しデータ、内部バスB4（以下アドレスバスとも記す）に読み出されているデータ又は内部バスB5（以下書き込みバスとも記す）に読み出されているデータと選択されたマスクパターンとが、マスク回路E5により論理積がとられ、その論理積が施されたデータ、すなわち、マスク処理（以下アライメント処理とも記す）が施されたデータが、レジスタ選択制御信号S7により選択されるアライメントレジスタAL0又はAL1に書き込まれる〔図12（13）〕。またアライメントレジスタAL0又はAL1を図6の応用例2のように中間バッファとして用いる場合、アライメントレジスタAL0又はAL1に書き込まれたデータを、さらに他のレジスタ、例えば汎用レジスタE8内の所定のレジスタ等に内部バスB1（以下ソースバスとも記す）・B2等を介して書き込む。

【0034】オペランドデコーダ11がデコードしたレジスタ番号が機能付きレジスタE3に含まれるレジスタであって、そのレジスタが前記シフトレジスタ（ $n_i$ ）ならば、データは、 $i$ に対応するビット数だけデータを左シフトするように、プリシフトE7が制御信号S1により制御される。すなわち、図12に示されるよう

に、命令デコードステージにおいてオペランドデコードが行われて当該シフトレジスタ $n_i$ の指定が解説されると、当該ステージにおいてレジスタ選択制御信号S7により選択される機能付きレジスタE3に含まれるシフトレジスタ $n_i$ からデータが内部バスB1に読出される〔図12（14）、（15）〕。次の演算ステージでは、オペランドデコードによって得られた制御信号S1によってプリシフトE7が制御されて、前記読み出しデータに対するシフト動作が行われる。シフトされたデータの処理は、当該シフトレジスタを指定した命令のオペレーションコードに従って処理される〔図12（16）〕。図8の応用例の場合だと、シフト処理されたデータは、算術論理演算器E9に入力される。そして演算ステージで算術論理演算器E9により、汎用レジスタE8内のレジスタbに格納されているデータと加算処理が施されてレジスタライトメモリストアステージで汎用レジスタE8内のレジスタtに、その加算処理結果のデータが格納される〔図12（16）、（17）〕。

【0035】また命令デコードステージでオペランドデコーダ11がデコードしたレジスタ番号が機能付きレジスタE3に含まれるインクリメントレジスタInの場合、レジスタ選択制御信号S7により選択される当該レジスタから読み出されたデータは内部バスB1及びプリシフトE7を通して算術論理演算器E9へ送られると同時に、内部バスB1を通して加算器E2へ送られる〔図11（3）、（5）〕。この加算器E2の他方の入力には、オペランドデコーダ11で解説されたインクリメントレジスタInの選択制御信号S3によってセレクトE1で選択された固定値+1が供給され、これによって、演算ステージで前記インクリメントレジスタInの読み出し値に1が加算される〔図11（8）〕。そしてレジスタライトメモリストアステージで、その格納されたデータが、書き込みバスB5を通じて再びインクリメントレジスタInへ書き戻される〔図11（9）〕。インクリメントレジスタInの値が次ステージで使用されるときは、制御ブロック14でそれを検出し、バイパス用ドライバDV0またはDV1を、制御信号S4またはS5をアサートすることによりイネーブル状態（開状態）にして、加算器E2の加算結果のデータをソースバスB1に直接出力しておく〔図1（10）〕。図3の応用例の場合、算術論理演算器E9には、インクリメントレジスタInの読み出しデータの他にプリシフトE7を介して、偏値dispが供給されている。算術論理演算器E9は、これを加算することにより、データキャッシュメモリ4のアドレス計算をする〔図11（6）〕。その後のステージ〔図11（4）、（7）〕については、モディファイドレジスタMの動作説明時に説明する。

【0036】命令デコードステージでオペランドデコーダ11がデコードしたレジスタ番号が機能付きレジスタE3に含まれる前記モディファイドレジスタMの場合、



選択制御信号S3がセクタE1の出力として偏値dispを選択して加算器E2の片方の入力に偏値dispを供給し、レジスタ選択制御信号S7により選択される当該レジスタのデータが内部バスB1を介して加算器E2の他方の入力に供給される〔図11(3)〕、

(5)〕。演算ステージで加算器E2は、モディファイドレジスタMのデータの値と偏値dispとを加算する〔図12(8)〕。そして、レジスタライトメモリストアステージで、その加算されたデータが、内部バスB5を通じて再びモディファイドレジスタMへ書き戻される〔図12(9)〕。次ステージでモディファイドレジスタMの値が使用されるときは制御ブロックI4でそれを検出し、バイパス用ドライバDV0またはDV1を、制御信号S9またはS5によりイネーブル状態にして、加算器E2の出力値を直接ソースバスB1へ出力しておく〔図11(10)〕。また、モディファイドレジスタMから読み出されたデータは、内部バスB1を介してプリシフタE7へも送られている。このとき、制御ブロックI4は当該偏値dispが負であるのか0又は正であるのかを判定し、さらにオペコードデコーダI2の解説結果に基づいて、オペレーションコードに規定される命令がLOAD命令か否かを判定する。そして制御ブロックI4は、当該偏値dispの判定結果と上記命令の判定結果に基づいて、制御信号S1をプリシフタE7に出力する。プリシフタE7は、算術論理演算器E9の片方の入力に、モディファイドレジスタMから読み出されたデータを供給する。さらに、上記命令の判定結果がLOAD命令の場合、プリシフタE7は、制御信号S1に基づいて、偏値dispが正又は0ならば0を負ならば偏値dispを選択して、算術論理演算器E9の他方の入力に供給する。尚、オペコードデコーダI2がデコードした命令が、LOAD命令で、そしてオペランドデコーダI1がデコードしたレジスタ番号がモディファイドレジスタM以外の場合、プリシフタE7は、制御信号S1に基づいて、アドレッシングの種類によってデータを選択し

(インデックスアドレッシング時はソースバスB1のデータを、偏値アドレッシング時は偏値dispを選択して)、算術論理演算器E9へ送る。演算ステージで、データキャッシュメモリ4のアドレスがモディファイドレジスタMから読み出されたデータと偏値disp又は0とを算術論理演算器E9が加算することにより計算される〔図11(6)〕。このアドレス計算は、図4の応用例を実現することになる。すなわち算術論理演算器E9の出力の値は、モディファイドレジスタMの読み出されたデータをM'として示すと、偏値dispが負の時" M' + disp"で、正又は0の時" M'"であり、応用例のメモリアクセスアドレスと内容的には同一である。算術論理演算器E9で計算されたアドレスは、アドレス信号として、アドレスバスB4へ送られ、これにより、データキャッシュメモリ4がそのアドレスに格納さ

れているデータを読み出しデータとして、マスク回路E5へ出力する〔図11(4)〕。レジスタメモリストアステージで、マスク回路E5は制御信号S6により制御され、ターゲットデコーダI3のデコードにより得られた書き込み先が汎用レジスタE8やアライメントレジスタ以外のオペレーティングレジスタの場合、マスク回路E5は、アライメント処理を実行せずに、上記書き込み先のレジスタに上記読み出しデータを書き込む。また、書き込み先がアライメントレジスタAL0ならば、マスク回路E5は、そのレジスタAL0に対応したマスクパターンのアライメント処理を上記読み出しデータに対して行い、アライメントレジスタAL0若しくはアライメントレジスタAL0を中間バッファとしてその他のレジスタヘデータを書き込む〔図11(7)〕。

【0037】オペランドデコーダI1がデコードしたレジスタ番号がモディファイドインデックスレジスタMiの場合、制御信号S3がセクタE1の出力としてソースバスB1を選択することにより、モディファイドインデックスレジスタMiの値のデータと任意のレジスタの値のデータとの加算が演算ステージにおいて加算器E2で行われる。以後の動作は、インクリメントレジスタInのときと同様である。

【0038】命令デコードステージで算術演算時のデータの書き込み先としてターゲットデコーダI3のデコードしたレジスタ番号がキャリー/ボローレジスタC/Bを指定すると、制御ブロックI4がそれを検出して、制御信号S2により、算術論理演算器E9の最下位ビットのキャリー入力として前回の演算時のキャリー又はボローの値PSW〔C/B〕のデータが選択される〔図12(18)〕。演算ステージで算術論理演算器E9は2つのデータとPSW〔C/B〕のデータを加算する〔図12(19)〕。その加算結果のデータは、内部バスB2を通してキャリー/ボローレジスタC/Bへ書き込まれる〔図12(20)〕。尚、データの書き込み先がキャリー/ボローレジスタC/B以外の場合は、セクタE6の出力は0が選ばれる。

【0039】図13には前記マスク回路E5の一例が示される。尚、図13中のS6a～S6cは、前記制御信号S6に含まれる制御信号であり、G0a～G23a, G1b～G23b及びG0c～G23cは、それぞれ制御信号図S6a～S6cにより開閉制御されるゲート回路である。図13に示される例は、レジスタのビット数が最大32ビットの場合であり、入力に対してマスクせずに出力する態様、入力の上位3バイト(ビット00～ビット23)をマスクして出力する態様、入力に上位ハーフワード(ビット00からビット16)をマスクして出力する態様を有し、各態様は制御信号S6a, S6b, S6cによって択一的に選択される。すなわち、ビット24～ビット31は入力から出力に至るスルーの信号経路を有する。ビット00～ビット23には、E5a



の領域に示されるように制御信号S 6 aで開閉制御されるゲート回路G 0 a～G 2 3 aを介して入力を選択的に出力に伝達する信号経路と、E 5 bの領域に示されるように制御信号S 6 bで開閉制御されるゲート回路G 0 b～G 2 3 bを介して論理値“0”の信号ビットを選択的に出力に伝達する信号経路とが設けられる。ビット0 0～ビット2 3には、E 5 c-1の領域に示されるように制御信号S 6 cで開閉制御されるゲート回路G 0 c～G 1 5 cを介して論理値“0”の信号ビットを選択的に出力に伝達する信号経路及びE 5 c-2の領域に示されるように制御信号S 6 dで開閉制御されるゲート回路G 1 6 c～G 2 3 cを介して入力を選択的に伝達する信号経路が設けられる。前記夫々のゲート回路は、特に制限されないが、クロックドインバータのような3ステート出力回路を含んで構成される。

【0040】ターゲットレジスタとしてアライメントレジスタAL 0及びAL 1が指定されない時（通常時）は、制御信号S 6 aのみアサートされ、E 5 aに含まれるゲート回路G 0 a～G 2 3 aがオン状態にされ、入力信号はそのまま対応するビットへ出力される。ターゲットレジスタとして、下位1バイトのみ“1”であるマスクパターンを持つアライメントレジスタAL 0又はAL 1が選択された時は、制御信号S 6 bのみがアサートされ、E 5 bに含まれるゲート回路G 0 b～G 2 3 bがオン状態にされ、入力信号の下位1バイトが、対応するビットに出力され、その他の上位3バイトには、“0”が出力される。ターゲットレジスタとして下位ハーフワードのみ“1”であるマスクパターンを持つアライメントレジスタAL 1又はAL 0が選択された時は、制御信号S 6 cのみがアサートされ、E 5 c-1及びE 5 c-2に含まれるゲート回路G 0 c～G 2 3 cがオン状態にされ、入力信号の下位ハーフワードが、対応するビットに出力され、その他の上位ハーフワードには、“0”が出力される。

【0041】図14には図9に示されるデータ処理装置のチップ平面が概略的に示される。同図において3は命令キャッシュメモリ（CC）、4はデータキャッシュメモリ（DC）、2は演算部（EU）、1は命令制御部（IU）、9はメモリ制御用ランダム論理（MU）及びシステムバス制御部（PU）、6は命令用タグキャッシュメモリ（CA）、7はデータ用タグキャッシュメモリ（DC）、5は命令用アドレス変換バッファ（CT）、7はデータ用アドレス変換バッファ（DT）、10は入出力部（I/O）である。

【0042】図15には機能付きレジスタをサポートしていないデータ処理装置と本発明に係る機能付きレジスタをサポートするデータ処理装置の、パイプラインステージで区切った動作フローチャートの一例が示される。機能付きレジスタをサポートしていない場合には、命令フェッチサイクルで図9の命令キャッシュメモリ3から

命令をフェッチし、デコードサイクルでフェッチした命令のオペコードデコードとオペランドデコードを行い、オペランドデコードによって示されるレジスタの内容を汎用レジスタから算術論理演算器に送る。演算サイクルでは、オペコードデコードで指示された演算を、フェッチしたレジスタの内容に対して算術論理演算器が行う。指示された演算がLOADもしくはSTORE命令ならば、続くメモリアクセスサイクルで、アクセスするメモリのアドレスと（STORE時は）書き込むデータをデータキャッシュメモリへ送る。そして、レジスタライト・メモリストアサイクルで、演算したデータを汎用レジスタまたはデータキャッシュメモリへ書き込む。命令がデータのアライメントを指示するものであれば、このサイクルでそれを行った後に書き込みを行う。

【0043】機能付きレジスタをサポートする場合には、デコードサイクルでデコードしたオペランドが、インクリメントレジスタIn、モディファイドレジスタM、インクリメントモディファイドレジスタMiのとき、次の演算ステージにてアドレスモディファイ・インクリメントを行う。また、デコードしたオペランドがシフテッドレジスタniの場合、演算サイクルで、演算・メモリアドレス計算を行う際に、あらかじめiビットだけデータをプリシフトする。

【0044】データの書き込み先のレジスタとして、アライメントレジスタAL 0又はAL 1、ノットレジスタNが指定されたときは、レジスタライト・メモリストアステージにてデータのマスクあるいはビット反転を行ってからデータ書き込みを行う。また書き込み先のレジスタがキャリア/ボローレジスタC/Bならば、演算ステージで演算を行う時に、キャリア/ボローの値を算術論理演算器E 9の最下位ビットに入力する。

【0045】上記実施例によれば以下の作用効果がある。

（1）本実施例に係る機能付きレジスタをサポートするデータ処理装置では、上述した多様なアドレッシングモードや複合命令等を、レジスタに機能を持たせることにより行う。即ち、命令数を増加させる代わりに機能付のレジスタを設け、オペコード領域のビット数の増加をオペランド領域とターゲット領域のビット数で吸収する。これに比べて、機能付きレジスタをサポートしていないデータ処理装置では、アドレッシングの方式を多様化したり幾つかの命令を組み合わせた複合命令を追加したりしてCISC的な機能を基本的な命令セットに持たせようとすると、命令数を増加することになり、命令フォーマット中、オペコードフィールドのビット数が増加してしまい、オペコードデコードの論理規模が大きくなり、命令デコード時間の増加、即ちデータ処理装置の動作周波数を低下させてしまうことになる。この相違は、図16からも明らかなように、本実施例データ処理装置では機能拡張に際してオペレーションコード領域のビット数

並びにオペコードデコーダの論理規模の増大は全くない。これに対して、機能付きレジスタをサポートしない従来方式ではオペレーションコード領域のビット数並びにオペコードデコーダの論理規模は著しく増大する。

(2) 更に、基本命令としてのオペレーションコードとオペランド保持領域としての機能付きレジスタとの組み合わせにより多様な機能を実現できるので、一つの機能付きレジスタを設ければ複数の命令若しくはオペレーションコードと組み合わせ使用できることになり、複数の機能を実現できる。例えば、ロード・アンド・インクリメントと、インクリメントと、ストア・アンド・インクリメントの機能を付加しようとするとき、従来方式では、それらに個々に対応した命令若しくはオペレーションコードを追加することになるが、本発明ではインクリメント・レジスタを1個追加するだけで済む。したがって、データ処理装置に追加した機能の数に比べて、追加すべき機能付きレジスタの数は比較的小数で済み、オペランド領域とターゲット領域のビット数の増加があまり大きくならない。したがって、機能付きレジスタによってデータ処理装置の機能を拡張しても、命令のオペランドやターゲットの指定領域のビット数、そしてオペランドデコーダ並びにターゲットデコーダの論理規模の増大を最小限に抑えることができる。

(3) 図2から図8までのオペレーティングレジスタを用いて、P A R i s c で定義される命令と同様な機能を実現した場合の命令数の削減を本発明者が試算したところ、例えば、全命令セット121命令のうち、前記各種機能付きレジスタを用いた効果により、26命令を削減できる。これは元の命令数の21%に相当する。

【0046】以上本発明者によってなされた発明を実施例に基づいて具体的に説明したが、本発明はそれに限定されるものではなく、その要旨を逸脱しない範囲において種々変更可能であることは言うまでもない。例えば、機能付きレジスタの種類は図2から図8で説明したものに限定されず、その他の機能を実現するようにもできる。以上の説明では主として本発明者によってなされた発明をその背景となった利用分野である汎用レジスタ方式のデータ処理装置に適用した場合について説明したが、本発明はそれに限定されるものではなく、アキュムレータ方式のデータ処理装置などにも適用可能である。また、専用の処理に専ら適用されるようなデータ処理装置の場合にも汎用レジスタを不要にすることができる。

【0047】

【発明の効果】本願において開示される発明のうち代表的なものによって得られる効果を簡単に説明すれば下記の通りである。

【0048】(1) レジスタに対するデータの書き込みまたは読み出し時に、そのデータに対して特定の処理を施す機能を予め所定のレジスタ(機能付きレジスタ)に

割当て、その機能付きレジスタを命令中に指定することにより、そのデータに対して、命令のオペレーションコードによって規定される命令の処理と機能付きレジスタの有する特定機能とを組み合わせた多様なデータ処理を実現することができる。換言すれば、命令数の増加を抑えて、多様なアドレッシングモードや複合命令による処理と同等の機能を実現できる。

(2) 命令のオペレーションコードによって規定される命令の処理と機能付きレジスタの有する特定機能とを組み合わせ多様なデータ処理を実現することにより、命令セットの命令数を増加することなしに、換言すれば、オペレーションコードのビット数を増大させることなく、データ処理装置による処理を多様化若しくは多機能化することができ、オペレーションコードのデコード論理に代表されるような命令制御部のハードウェアの増大を極力抑えて多様な機能を実現できる。これにより更に、命令制御部の命令デコーダ等のランダムロジックの論理規模が大きくなりすぎずに済み、LSIの性能低下も防止できる。

(3) 上記によって実現される多様な機能は一つの命令中のオペランド指定領域などでの機能付きレジスタの指定で行われ、且つ、その機能付きレジスタに割当てられる固有の機能は当該レジスタに対するデータの書き込みまたは読み出し時に行われるので、機能拡張に伴って実行すべき命令数が増えず、このことは、高機能な処理を少ないステップ数を以て実現できることも意味する。

(4) 基本命令としてのオペレーションコードとオペランド保持領域としての機能付きレジスタとの組み合わせにより多様な機能を実現できるので、一つの機能付きレジスタを設ければ複数の命令若しくはオペレーションコードと組み合わせ使用できることになり、複数の機能を実現できる。したがって、データ処理装置に追加した機能の数に比べて、追加すべき機能付きレジスタの数は比較的小数で済み、オペランド領域とターゲット領域のビット数の増加があまり大きくならない。これにより、機能付きレジスタによってデータ処理装置の機能を拡張しても、命令のオペランドやターゲットの指定領域のビット数、そしてオペランドデコーダ並びにターゲットデコーダの論理規模の増大を最小限に抑えることができる。

【図面の簡単な説明】

【図1】本発明に係るデータ処理装置の機能付きレジスタの定義と機能の一例説明図である。

【図2】機能付きレジスタの一例であるノット・レジスタのC言語的な定義並びにそれを用いた処理例を概念的に示す説明図である。

【図3】機能付きレジスタの一例であるインクリメントレジスタのC言語的な定義並びにそれを用いた処理例を概念的に示す説明図である。

【図4】機能付きレジスタの一例であるモディファイド

レジスタのC言語的な定義並びにそれを用いた処理例を概念的に示す説明図である。

【図5】機能付きレジスタの一例であるモディファイドインデックスレジスタ並びにアライメントレジスタのC言語的な定義並びにそれを用いた処理例を概念的に示す説明図である。

【図6】前記アライメントレジスタを用いた別の処理例を示す説明図である。

【図7】機能付きレジスタの一例であるキャリー／ポローレジスタのC言語的な定義並びにそれを用いた処理例を概念的に示す説明図である。

【図8】機能付きレジスタの一例であるシフテッドレジスタのC言語的な定義並びにそれを用いた処理例を概念的に示す説明図である。

【図9】図2から図8で説明した機能付きレジスタを用いたデータ処理装置の一実施例ブロック図である。

【図10】機能付きレジスタ及び汎用レジスタの指定方式を示す一例説明図である。

【図11】機能付きレジスタのオペレーションを図9の構成に即して実行するときの制御方式をパイプライン形式で示す一例説明図である。

【図12】図11とは別の機能付きレジスタのオペレーションを図9の構成に即して実行するときの制御方式をパイプライン形式で示す一例説明図である。

【図13】マスク回路の一例回路図である。

【図14】図9に示されるデータ処理装置のチップ平面図である。

【図15】機能付きレジスタを有しないデータ処理装置と本発明に係るデータ処理装置の、パイプラインステージで区切った一例動作フローチャートである。

【図16】命令のフォーマットとハードウェアとの関係を示した説明図である。

【符号の説明】

N ノットレジスタ  
 In インクリメントレジスタ  
 M モディファイドレジスタ  
 Mi モディファイドインデックスレジスタ  
 AL0 アライメントレジスタ  
 Ag0 アライメントレジスタ（中間バッファ）  
 C/B キャリー／ポローレジスタ  
 ni シフテッドレジスタ  
 1 命令制御部  
 I1 オペランドデコーダ  
 I2 オペコードデコーダ  
 I3 ターゲットデコーダ  
 I4、I5 その他の制御ブロック  
 2 演算部  
 E1 セレクタ  
 E2 加算器  
 E3 機能付きレジスタ  
 E4 反転回路  
 E5 マスク回路  
 E6 セレクタ  
 E7 プリシフタ  
 E8 汎用レジスタ  
 E9 算術論理演算器  
 E10 プログラムカウンタ  
 B3～B0 レジスタ指定情報  
 B3 汎用レジスタ／機能付きレジスタ指定ビット

【図2】

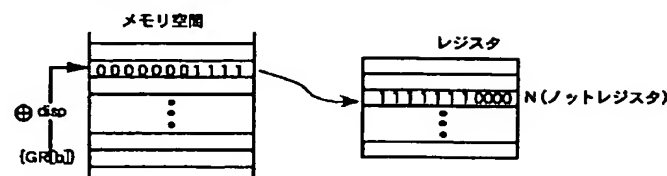
【図2】

## 1. ノットレジスタ

定義 if write data is written to Reg[N]  
 Reg[N] ← complement (write data);

応用例

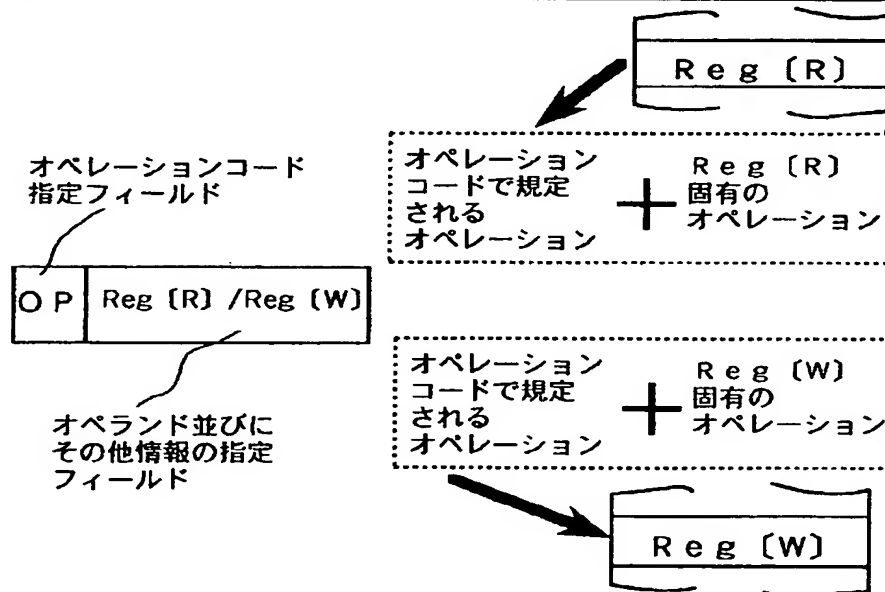
LOAD disp(b),N



【図 1】

【図 1】

- (1) Operating read data Register  
 if read data is read from Reg[R]  
 $\{ \text{read data} \leftarrow \text{Fr}(i) (\text{Reg}[R]) \}$  andor  $\{ \text{Reg}[R] \leftarrow \text{Fr}(i) (\text{Reg}[R]) \};$
- Fr (i) : i番目の命令の特定フィールドの値をパラメータとする  
 レジスタ R 固有の演算子
- (2) Operating write data Register  
 if write data is written to Reg[W]  
 $\text{Reg}[W] \leftarrow \text{Fw}(i) (\text{Reg}[W]);$
- Fw (i) : i番目の命令の特定フィールドの値をパラメータとする  
 レジスタ W 固有の演算子



【図 3】

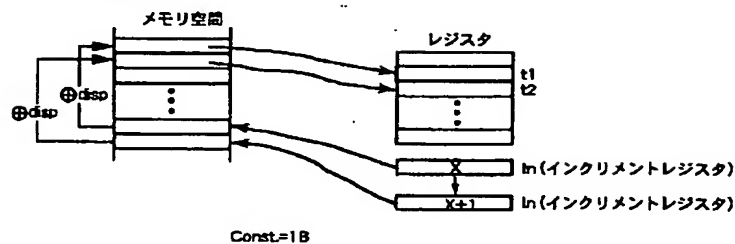
【図 3】

## 2. インクリメントレジスタ

定義 if read data is read from Reg[In]  
 $Reg[In] \leftarrow Reg[In] + Const.$

応用例

LOAD disp(In), t1  
 LOAD disp(In), t2



【図 4】

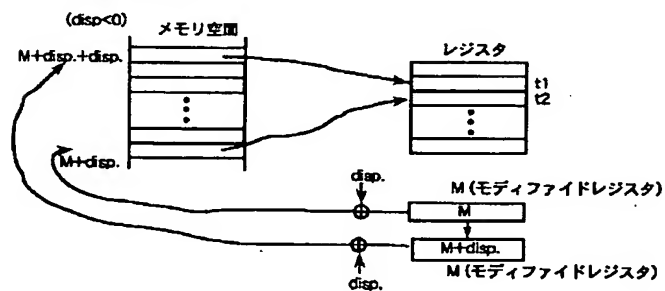
【図 13】

## 3. モディファイドレジスタ

定義 if read data is read from Reg[M] {  
 if disp() < 0  
 read data  $\leftarrow Reg[M]$   
 else  
 read data  $\leftarrow Reg[M] - disp.$   
 }  
 $Reg[M] \leftarrow Reg[M] + disp.()$

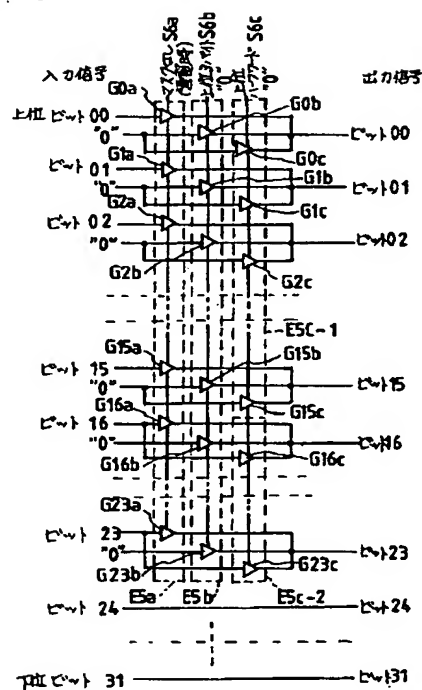
応用例

LOAD disp(M), t1  
 LOAD disp(M), t2



【図 4】

【図 13】



【図 5】

4. モディファイドインデックスレジスタ

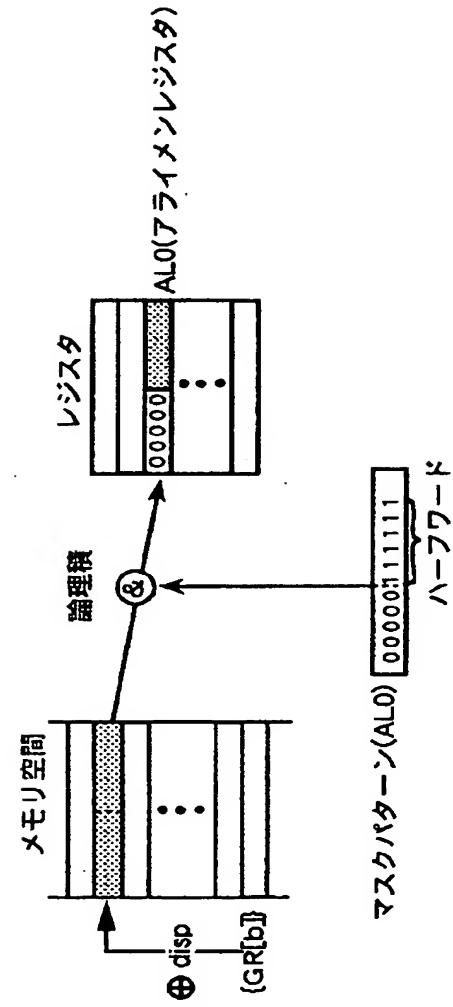
定義 if read data is read from Reg[Mi]  
 $\text{Reg}[\text{Mi}] \leftarrow \text{Reg}[\text{Mi}] + \text{Reg}[\text{r}](i);$

5. アライメントレジスタ

定義 if write data is written to Reg[AL] (i)  
 $\text{Reg}[\text{AL}] \leftarrow \text{write data} \& \text{maskpattern}(\text{AL}(i));$

応用例 1

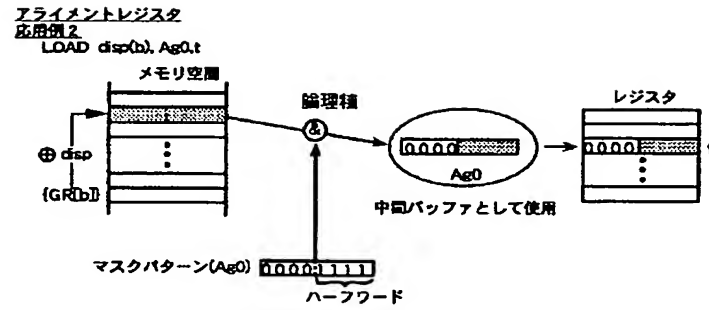
LOAD disp(b), A10



【図 5】

【図 6】

【図 6】



【図 7】

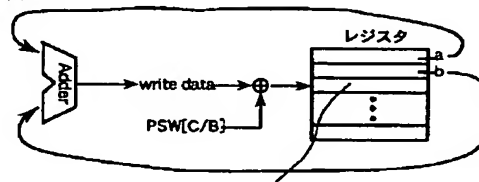
【図 7】

#### 6. キャリー／ボローレジスタ

定義 if write data is written to Reg[C/B]  
Reg[C/B] ← write data + PSW[C/B]

応用例

ADD a, b, C/B



C/B (キャリー／ボローレジスタ)

【図 8】

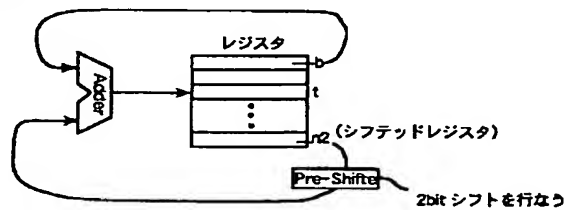
【図 8】

#### 7. シフトレジスタ

定義 if read data is read from Reg[ni]  
read data ← shift (Reg[ni], 0);

応用例

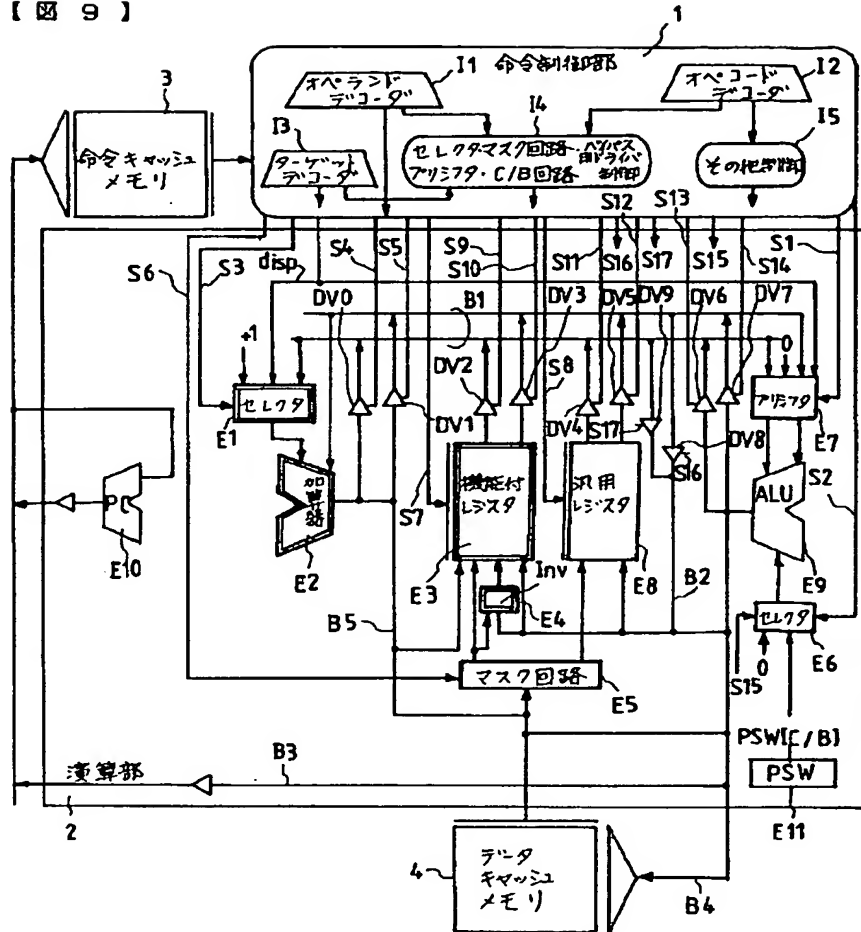
ADD n2, b, t





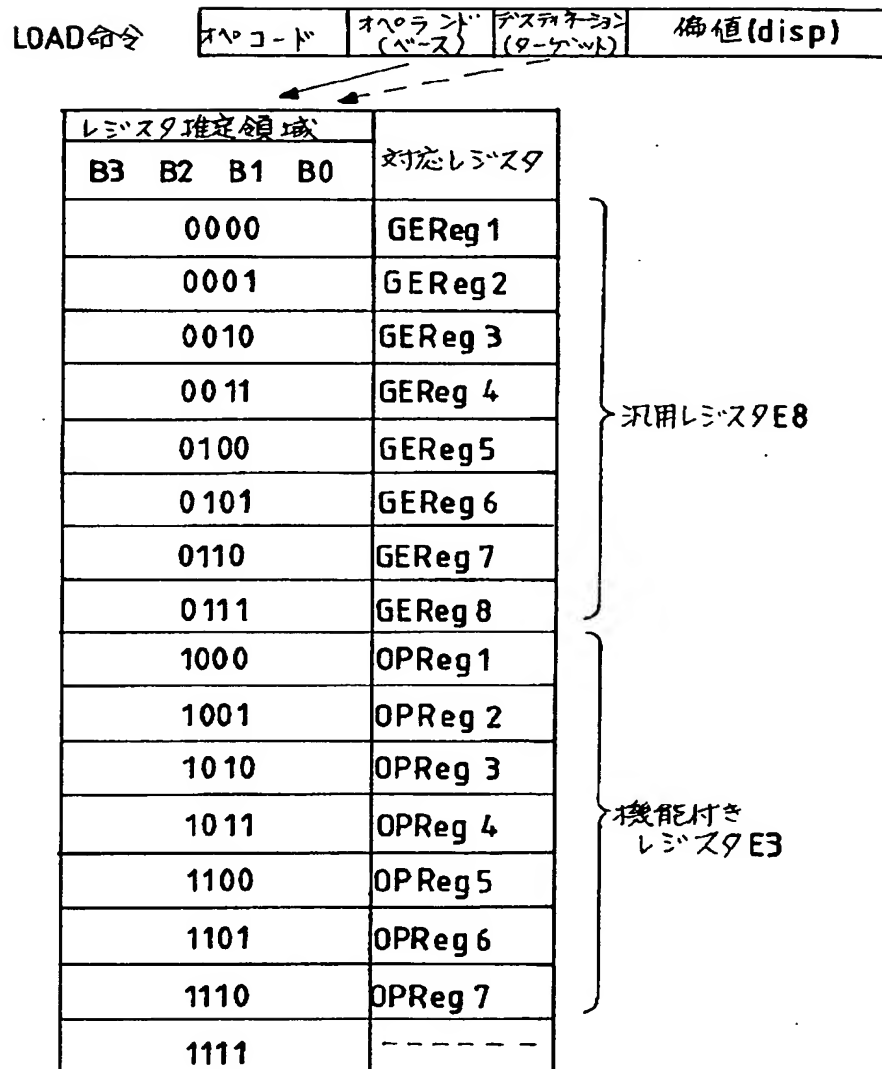
【図 9】

【図 9】



【図 10】

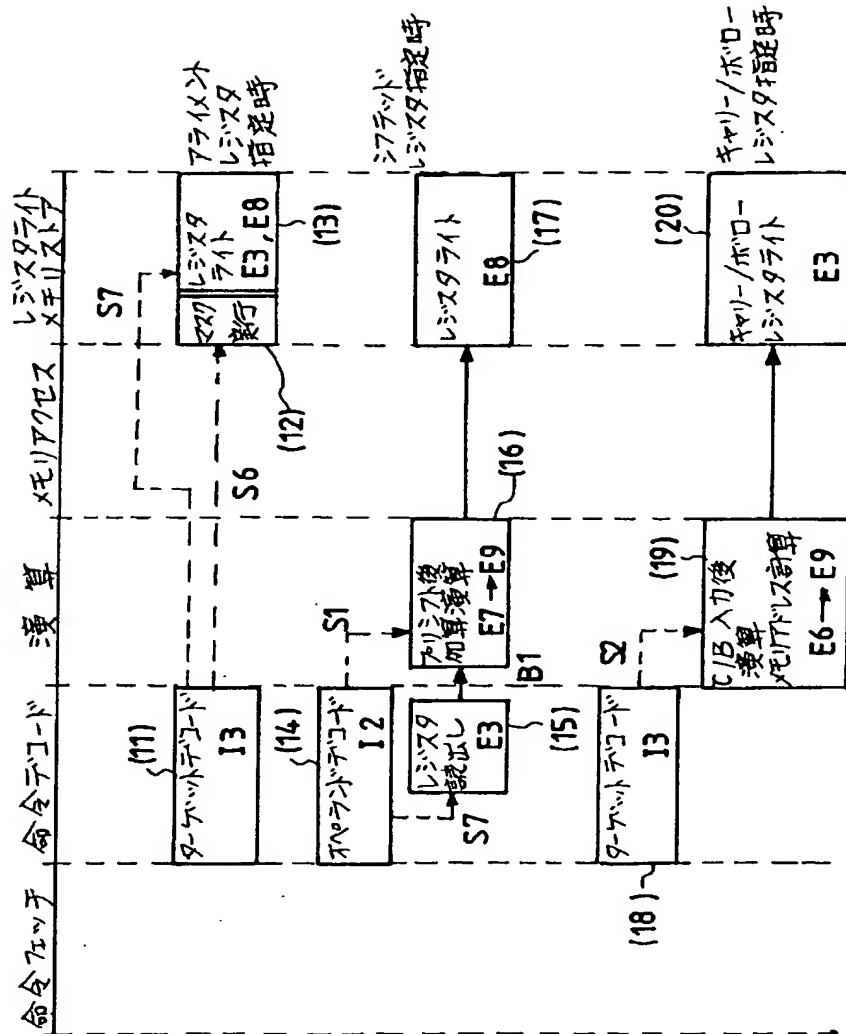
【図 10】





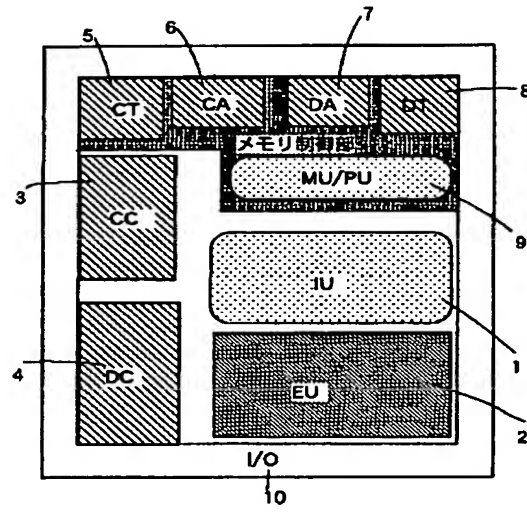
【図 12】

【図 12】



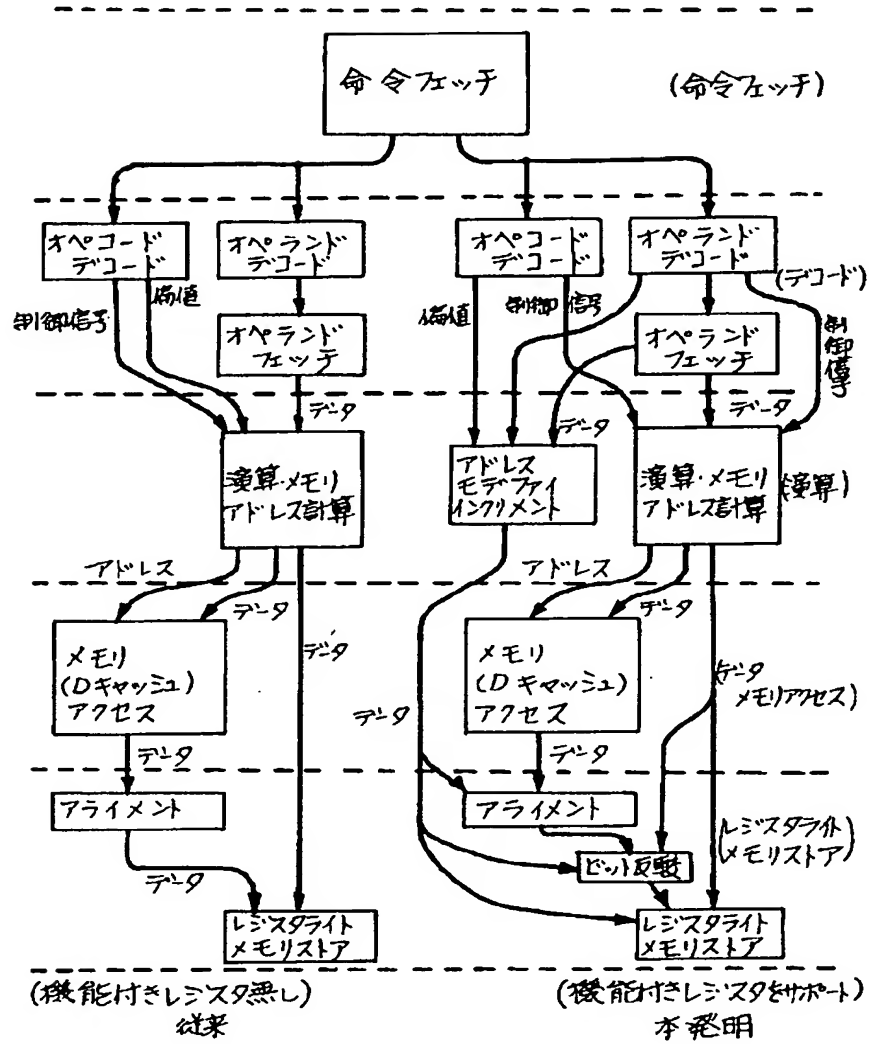
【図14】

【図14】



【図15】

【図 15】



【図16】

【図16】

	従来方式	本発明
オペコード領域のビット数	×	○
オペランド領域のビット数	○	△
ターゲット領域のビット数	○	△
オペコードデコーダの論理規模	×	○
オペランドデコーダの論理規模	○	△
ターゲットデコーダの論理規模	○	△

○ 増加無し  
△ 増加小  
× 増加大